Mathe Artikel

Ein kleiner Beitrag zur Mathematik: Die Begleitmatrix

Ermittlung des charakteristischen Polynoms durch bilaterale Elimination

Sergej Oks

Mathe Artikel

Ein kleiner Beitrag zur Mathematik: Die Begleitmatrix

Ermittlung des charakteristischen Polynoms durch bilaterale Elimination

Sergej Oks

20. Dezember 2025

Inhaltsverzeichnis

1 Intention **1**

2 Bilaterale Elimination **3**
2.1 Konstruktion der Transformationsmatrix 4
2.2 Lösungsverfahren 6
2.3 Eine kurze Beispielberechnung 9

3 Fazit **11**

4 Anhang **13**
4.1 Matlab-Code: Begleitmatrix (original) 13
4.2 Matlab-Code: Begleitmatrix (optimiert) 14
4.3 Matlab-Code: Begleitmatrix (Danilewski) 15

Literatur und Anmerkung zur Erstellung **16**

1 Intention

Diese Arbeit soll die Möglichkeit der Ermittlung des charakteristischen Polynoms $p_A(\lambda)$ durch Transformation einer quadratischen Matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ in eine sogenannte Begleitmatrix $\tilde{\boldsymbol{A}}$ dokumentieren. Es geht hier explizit nicht um die Berechnung der zugehörigen Eigenwerte λ_i mit $i = 1$ bis n. Die Begleitmatrix ist so aufgebaut, dass in der Subdiagonalen Einsen stehen und in der letzten Spalte die (negativen) Koeffizienten des charakteristischen Polynoms.

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \tag{1.1}$$

$$\Rightarrow \tilde{\boldsymbol{A}} = \text{transform}(\boldsymbol{A}) = \begin{bmatrix} 0 & 0 & \dots & & \dots & 0 & \tilde{A}_{1n} \\ 1 & 0 & \dots & & \dots & 0 & \tilde{A}_{2n} \\ 0 & 1 & \ddots & & & \vdots & \vdots \\ \vdots & \ddots & \ddots & & & & \\ & & & & & & \\ \vdots & & & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & & \ddots & 1 & 0 & \tilde{A}_{(n-1)n} \\ 0 & 0 & \dots & \dots & 0 & 1 & \tilde{A}_{nn} \end{bmatrix} \tag{1.2}$$

Diese ergeben sich bei der Berechnung automatisch. Es sei angemerkt, dass man alternativ auch eine transponierte Begleitmatrix ermitteln kann, in der die Koeffizienten dementsprechend in der letzten Zeile sichtbar werden würden. Je nach Literatur gibt es unterschiedliche Darstellungsweisen.

Das charakteristische Polynom ergibt sich hier aus $\text{Det}\,(\lambda \boldsymbol{I} - \boldsymbol{A}) = \text{Det}\,(\lambda \boldsymbol{I} - \tilde{\boldsymbol{A}})$ zu:

$$p_A(\lambda) = p_{\tilde{A}}(\lambda) = \lambda^n - \tilde{A}_{nn}\lambda^{n-1} - \tilde{A}_{(n-1)n}\lambda^{n-2} - \dots - \tilde{A}_{2n}\lambda^1 - \tilde{A}_{1n}\lambda^0 \tag{1.3}$$

Und durch Nullsetzen des Polynoms $p_A(\lambda) = p_{\tilde{A}}(\lambda) = 0$ erhält man die charakteristische Gleichung zur Bestimmung der Eigenwerte:

$$\lambda^n = \tilde{A}_{nn}\lambda^{n-1} + \tilde{A}_{(n-1)n}\lambda^{n-2} + \dots + \tilde{A}_{2n}\lambda^1 + \tilde{A}_{1n}\lambda^0 \tag{1.4}$$

Für die Ermittlung der Begleitmatrix ist es notwendig, die grundlegenden Eigenschaften

einer Matrix während ihrer Transformation beizubehalten. Konkret geht es um die Erhaltung der Spur und somit der Summe aller Eigenwerte, sowie der Determinante und somit des Produkts aller Eigenwerte. Insbesondere muss also Folgendes gelten:

$$\text{Spur}\,(\boldsymbol{A}) = \text{Spur}\,(\tilde{\boldsymbol{A}}) = \tilde{A}_{nn} \tag{1.5}$$

Und unter Berücksichtigung des alternierenden Vorzeichens je nach Dimension n:

$$\text{Det}\,(\boldsymbol{A}) = \text{Det}\,(\tilde{\boldsymbol{A}}) = (-1)^{n+1}\tilde{A}_{1n} \tag{1.6}$$

Die Invarianz dieser beiden Eigenschaften ist bei der sogenannten Ähnlichkeitstransformation gegeben. Wobei, wie bereits erwähnt, auf eine explizite Berechnung der Eigenwerte verzichtet wird. Dies ist mit dem sogenannten Danilewski-Verfahren [1] möglich, welches in etwas abgewandelter Form im folgenden Kapitel besprochen werden soll.

2 Bilaterale Elimination

Die bilaterale Elimination wird durch schrittweise Anwendung von Ähnlichkeitstransformationen erreicht. Während bei einer unilateralen Elimination (wie z.B. beim Gaußschen Eliminationsverfahren) eine einseitige Multiplikation mit einer Elementarmatrix vorgenommen wird, um entweder Zeilen- oder Spaltenoperationen durchzuführen, wird bei der bilateralen Elimination (also wie beim Danilewski-Verfahren [1]) jede Zeilenoperation durch eine korrespondierende Spaltenoperation ausgeglichen. Konkret ergibt sich eine Zeilenoperation aus einer linksseitigen Multiplikation

$$\tilde{\boldsymbol{A}}_{Zeilen} = \boldsymbol{L} \cdot \boldsymbol{A} \tag{2.1}$$

und eine Spaltenoperation aus einer rechtsseitigen Multiplikation.

$$\tilde{\boldsymbol{A}}_{Spalten} = \boldsymbol{A} \cdot \boldsymbol{R} \tag{2.2}$$

Die beidseitige Multiplikation bei einer Ähnlichkeitstransformation manipuliert also sowohl Zeilen als auch Spalten. Was es etwas schwieriger macht, eine geeignete Transformationsmatrix $\boldsymbol{P}$ zu finden, mit der man Nullen erzeugen kann.

$$\tilde{\boldsymbol{A}} = \boldsymbol{P}^{-1} \cdot \boldsymbol{A} \cdot \boldsymbol{P} \tag{2.3}$$

Des Weiteren folgt durch die Invertierung der Transformationsmatrix auf einer der beiden Seiten zusätzlich die Umkehrung der entsprechenden Operation. Eine Addition (bzw. Bildung einer Summe) zieht also eine Subtraktion (bzw. Bildung von Differenzen) nach sich, was zur Spurerhaltung führt und eine Multiplikation wird verbunden mit einer Division, was die Determinantenerhaltung garantiert. Wie im Kapitel 1 bereits klargestellt, sind diese Eigenschaften essentiell für die Erhaltung der Eigenwerte und somit des charakteristischen Polynoms der Matrix $\boldsymbol{A}$. Die Invarianz kann man gut erkennen, wenn man die Gleichung 2.3 umstellt.

$$\Leftrightarrow \boldsymbol{P} \cdot \tilde{\boldsymbol{A}} = \boldsymbol{A} \cdot \boldsymbol{P} \tag{2.4}$$

Es wird recht leicht klar, dass die Matrizen $\boldsymbol{A}$ und $\tilde{\boldsymbol{A}}$ im Wesentlichen gleich sein müssen und sich lediglich, infolge der unterschiedlichen Anwendung der Transformationsmatrix $\boldsymbol{P}$ (linksseitig als Zeilenoperation und rechtsseitig als Spaltenoperation), in ihrer Erscheinung unterscheiden.

2.1 Konstruktion der Transformationsmatrix

Die Transformationsmatrix $\boldsymbol{P}$ für eine Stelle x in $i = 1$ bis n kann gebildet werden aus einer Elementarmatrix für die Multiplikation

$$\boldsymbol{P}_{mult} = \begin{bmatrix} P_{11} & 0 & \dots & & \dots & & \dots & 0 & 0 \\ 0 & P_{22} & \ddots & & & & & 0 & 0 \\ \vdots & \ddots & \ddots & & & & & & \vdots \\ & & & P_{(x-1)(x-1)} & 0 & & & & \\ \vdots & & & 0 & P_{xx} & 0 & & & \vdots \\ & & & & 0 & P_{(x+1)(x+1)} & & & \\ \vdots & & & & & & \ddots & \ddots & \vdots \\ 0 & 0 & & & & & \ddots & P_{(n-1)(n-1)} & 0 \\ 0 & 0 & \dots & & \dots & & \dots & 0 & P_{nn} \end{bmatrix} \tag{2.5}$$

oder Division

$$\boldsymbol{P}_{div} = \boldsymbol{P}_{mult}^{-1} = \begin{bmatrix} \frac{1}{P_{11}} & 0 & \dots & & \dots & & \dots & 0 & 0 \\ 0 & \frac{1}{P_{22}} & \ddots & & & & & 0 & 0 \\ \vdots & \ddots & \ddots & & & & & & \vdots \\ & & & \frac{1}{P_{(x-1)(x-1)}} & 0 & & & & \\ \vdots & & & 0 & \frac{1}{P_{xx}} & 0 & & & \vdots \\ & & & & 0 & \frac{1}{P_{(x+1)(x+1)}} & & & \\ \vdots & & & & & & \ddots & \ddots & \vdots \\ 0 & 0 & & & & & \ddots & \frac{1}{P_{(n-1)(n-1)}} & 0 \\ 0 & 0 & \dots & & \dots & & \dots & 0 & \frac{1}{P_{nn}} \end{bmatrix} \tag{2.6}$$

in Kombination mit einer Elementarmatrix für die einfache Addition mit dem Einsvektor an der Stelle x

$$\boldsymbol{P}_{add} = \begin{bmatrix} 1 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \ddots & 0 & 1 & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ & & & 1 & 1 & 0 & & & \\ \vdots & & & 0 & 1 & 0 & & & \vdots \\ & & & 0 & 1 & 1 & & & \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & 1 & 0 & \ddots & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 1 \end{bmatrix} \tag{2.7}$$

oder der entsprchenden Elementarmarix für die Differenz.

$$\boldsymbol{P}_{diff} = \boldsymbol{P}_{add}^{-1} = \begin{bmatrix} 1 & 0 & \dots & 0 & -1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \ddots & 0 & -1 & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ & & & 1 & -1 & 0 & & & \\ \vdots & & & 0 & 1 & 0 & & & \vdots \\ & & & 0 & -1 & 1 & & & \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & -1 & 0 & \ddots & 1 & 0 \\ 0 & 0 & \dots & 0 & -1 & 0 & \dots & 0 & 1 \end{bmatrix} \tag{2.8}$$

Die hier aufgeführten Matrizen für jeweils einen separaten Umwandlungsschritt lassen sich prinzipiell unterschiedlich (auch transponiert) zusammensetzen, wobei man drauf achten muss, dass die Reihenfolge der Operationen an bestimmten Stellen Nullen erzeugen soll und somit zuerst eine Multiplikation/Division und danach eine Addition/Subtraktion erfolgen muss (wie beim Gaußverfahren). In dieser Arbeit soll die folgende Kombination ausgewählt werden

$$\boldsymbol{P} = \boldsymbol{P}_{mult} \cdot \boldsymbol{P}_{add} \tag{2.9}$$

$$\Rightarrow \boldsymbol{P} = \begin{bmatrix} P_{11} & 0 & \dots & 0 & P_{11} & 0 & \dots & 0 & 0 \\ 0 & P_{22} & \ddots & 0 & P_{22} & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ \vdots & & & P_{(x-1)(x-1)} & P_{(x-1)(x-1)} & 0 & & & \vdots \\ \vdots & & & 0 & P_{xx} & 0 & & & \vdots \\ \vdots & & & 0 & P_{(x+1)(x+1)} & P_{(x+1)(x+1)} & & & \vdots \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & P_{(n-1)(n-1)} & 0 & \ddots & P_{(n-1)(n-1)} & 0 \\ 0 & 0 & \dots & 0 & P_{nn} & 0 & \dots & 0 & P_{nn} \end{bmatrix} \tag{2.10}$$

und somit auch die entsprechende Inverse.

$$\boldsymbol{P}^{-1} = \begin{bmatrix} \frac{1}{P_{11}} & 0 & \dots & 0 & -\frac{1}{P_{xx}} & 0 & \dots & 0 & 0 \\ 0 & \frac{1}{P_{22}} & \ddots & 0 & -\frac{1}{P_{xx}} & 0 & & 0 & 0 \\ \vdots & \ddots & \ddots & & \vdots & & & & \vdots \\ \vdots & & & \frac{1}{P_{(x-1)(x-1)}} & -\frac{1}{P_{xx}} & 0 & & & \vdots \\ \vdots & & & 0 & \frac{1}{P_{xx}} & 0 & & & \vdots \\ \vdots & & & 0 & -\frac{1}{P_{xx}} & \frac{1}{P_{(x+1)(x+1)}} & & & \vdots \\ \vdots & & & & \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & & 0 & -\frac{1}{P_{xx}} & 0 & \ddots & \frac{1}{P_{(n-1)(n-1)}} & 0 \\ 0 & 0 & \dots & 0 & -\frac{1}{P_{xx}} & 0 & \dots & 0 & \frac{1}{P_{nn}} \end{bmatrix} \tag{2.11}$$

Die gebildete Transformationsmatrix beinhaltet eine Diagonale und einen Vektor an der Stelle x mit den Faktoren $P_{ii} \neq 0$ für $i = 1$ bis n. Alle anderen Einträge sind Null. Die Einfachheit der Inversen ermöglicht es, sie direkt zu erstellen, ohne sie algorithmisch ermitteln zu müssen. Was den Rechenaufwand verringert.

2.2 Lösungsverfahren

Die im vorrangegangenen Kapitel erläuterte Transformationsmatrix $\boldsymbol{P}$ kann nun genutzt werden, um für eine gegebene Matrix

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ A_{31} & A_{32} & \dots & A_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \tag{2.12}$$

schrittweise eine Lösung zu ermitteln. Aus den Komponenten A_{ik} erfolgt im ersten Schritt $k = 1$ die Übertragung der Einträge des Vektors $\boldsymbol{A}_{i1}$ in die entsprechenden Positionen P_{ii} in der Transformationsmatrix. Dabei enthält $\boldsymbol{P}$ zusätzlich zu den Werten in der Hauptdiagonalen den vollständigen Vektor $\boldsymbol{A}_{i1}$ an der Stelle $x = k + 1 = 2$.

$$\boldsymbol{P} = \begin{bmatrix} A_{11} & A_{11} & 0 & 0 & \dots & 0 \\ 0 & A_{21} & 0 & 0 & \dots & 0 \\ 0 & A_{31} & A_{31} & 0 & \dots & 0 \\ 0 & A_{41} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & A_{(n-1)1} & 0 \\ 0 & A_{n1} & 0 & \dots & 0 & A_{n1} \end{bmatrix} \tag{2.13}$$

Die zugehörige Inverse ergibt sich zu:

$$\boldsymbol{P}^{-1} = \begin{bmatrix} 1/A_{11} & -1/A_{21} & 0 & 0 & \dots & 0 \\ 0 & 1/A_{21} & 0 & 0 & \dots & 0 \\ 0 & -1/A_{21} & 1/A_{31} & 0 & \dots & 0 \\ 0 & -1/A_{21} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1/A_{(n-1)1} & 0 \\ 0 & -1/A_{21} & 0 & \dots & 0 & 1/A_{n1} \end{bmatrix} \tag{2.14}$$

Die Anwendung dieser Transformation führt dazu, dass im ersten Schritt zwar die erste Spalte $k = 1$ der Matrix $\boldsymbol{A}$ bearbeitet wird, jedoch der einzige Eintrag ungleich Null in der zweiten Zeile dieser Spalte übrig bleibt. Schließlich geht es darum, eine Subdiagonale zu bilden, was den Startwert $x = 2$ impliziert und woraus folgt, dass der Lösungsalgorithmus maximal $(n-1)$-Schritte durchlaufen kann. Falls nicht sofort ersichtlich ist, dass der Startwert nicht $x = 1$ sein kann, muss man sich vergegenwärtigen, dass es hier nicht um die Hauptdiagonale geht. Diese enthielte nämlich die Eigenwerte, die aufgrund ihrer nicht-linearen Eigenschaften nicht auf diese relativ einfache Art und Weise ermittelt werden können. Man erhält

$$\boldsymbol{A}' = \boldsymbol{P}^{-1} \cdot \boldsymbol{A} \cdot \boldsymbol{P} = \begin{bmatrix} 0 & A'_{12} & \dots & A'_{1n} \\ A'_{21} & A'_{22} & \dots & A'_{2n} \\ 0 & A'_{32} & \dots & A'_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & A'_{n2} & \dots & A'_{nn} \end{bmatrix} \tag{2.15}$$

und kann mit dem nächsten Schritt $k = 2$ und somit $x = 3$ weitermachen. Die dazugehörige Transformationsmatrix ergibt sich zu

$$\boldsymbol{P}' = \begin{bmatrix} A'_{12} & 0 & A'_{12} & 0 & 0 & \dots & 0 \\ 0 & A'_{22} & A'_{22} & 0 & 0 & \dots & 0 \\ 0 & 0 & A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 0 & A'_{42} & A'_{42} & 0 & \dots & 0 \\ 0 & 0 & A'_{52} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & A'_{(n-1)2} & 0 \\ 0 & 0 & A'_{n2} & 0 & \dots & 0 & A'_{n2} \end{bmatrix} \tag{2.16}$$

mit der entsprechenden Inversen

$$\boldsymbol{P'}^{-1} = \begin{bmatrix} 1/A'_{12} & 0 & -1/A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 1/A'_{22} & -1/A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 0 & 1/A'_{32} & 0 & 0 & \dots & 0 \\ 0 & 0 & -1/A'_{32} & 1/A'_{42} & 0 & \dots & 0 \\ 0 & 0 & -1/A'_{32} & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & 1/A'_{(n-1)2} & 0 \\ 0 & 0 & -1/A'_{32} & 0 & \dots & 0 & 1/A'_{n2} \end{bmatrix} \tag{2.17}$$

sodass mit dem Ergebnis

$$\boldsymbol{A''} = \boldsymbol{P'}^{-1} \cdot \boldsymbol{A'} \cdot \boldsymbol{P'} = \begin{bmatrix} 0 & 0 & A''_{13} & \dots & A''_{1n} \\ A''_{21} & 0 & A''_{23} & \dots & A''_{2n} \\ 0 & A''_{32} & A''_{33} & \dots & A''_{3n} \\ 0 & 0 & A''_{43} & \dots & A''_{4n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & A''_{n3} & \dots & A''_{nn} \end{bmatrix} \tag{2.18}$$

weitere Schritte bis zum $(n-1)$-Schritt

$$\boldsymbol{A}^{(n-1)} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & A_{1n}^{(n-1)} \\ A_{21}^{(n-1)} & 0 & 0 & \dots & 0 & A_{2n}^{(n-1)} \\ 0 & A_{32}^{(n-1)} & 0 & \dots & 0 & A_{3n}^{(n-1)} \\ 0 & 0 & A_{43}^{(n-1)} & \ddots & 0 & A_{4n}^{(n-1)} \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & A_{n(n-1)}^{(n-1)} & A_{nn}^{(n-1)} \end{bmatrix} \tag{2.19}$$

erfolgen können. Abschließend müssen die Einträge in der Subdiagonale noch geteilt werden, sodass dort nur noch Einsen stehen. Da jede Multiplikation oder Division in einer Zeile auch gleichzeitig die umgekehrte Wirkung auf die entsprechende Spalte hat, sind immer jeweils zwei Subdiagonalelemente miteinander verbunden. Daraus folgt eine sukzessive Anwendung der Operationen $P_{ii} = P_{(i-1)(i-1)} \cdot A_{i(i-1)}$ von $i = 2$ bis n, sodass sich mit dem Startwert $P_{11} = 1$ folgende Transformationsmatrix ergibt.

$$\boldsymbol{P}^{(n-1)} = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & A_{21}^{(n-1)} & 0 & 0 & \dots & 0 \\ 0 & 0 & A_{21}^{(n-1)} \cdot A_{32}^{(n-1)} & 0 & \dots & 0 \\ 0 & 0 & 0 & A_{21}^{(n-1)} \cdot A_{32}^{(n-1)} \cdot A_{43}^{(n-1)} & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & A_{21}^{(n-1)} \cdot \dots \cdot A_{n(n-1)}^{(n-1)} \end{bmatrix} \tag{2.20}$$

Die Begleitmatrix berechnet man wie gewohnt mit

$$\boldsymbol{A}^{(n)} = (\boldsymbol{P}^{(n-1)})^{-1} \cdot \boldsymbol{A}^{(n-1)} \cdot \boldsymbol{P}^{(n-1)} \tag{2.21}$$

und erhält sie in der endgültigen Form

$$\Leftrightarrow \tilde{\boldsymbol{A}} = \boldsymbol{A}^{(n)} = \begin{bmatrix} 0 & 0 & \dots & & \dots & 0 & \tilde{A}_{1n} \\ 1 & 0 & \dots & & \dots & 0 & \tilde{A}_{2n} \\ 0 & 1 & \ddots & & & \vdots & \vdots \\ \vdots & \ddots & \ddots & & & & \\ & & & & & & \\ \vdots & & & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & & \ddots & 1 & 0 & \tilde{A}_{(n-1)n} \\ 0 & 0 & \dots & \dots & 0 & 1 & \tilde{A}_{nn} \end{bmatrix} \tag{2.22}$$

sodass man hier das charakteristische Polynom, wie in Kapitel 1 beschrieben, ablesen kann.

2.3 Eine kurze Beispielberechnung

Zur Veranschaulichung soll in diesem Abschnitt ein verkürztes 3×3 Beispiel durchgerechnet werden, wobei der erste Schritt im Prinzip bereits im vorrangegangenen Abschnitt offensichtlich ist. Es reicht aus, die ersten drei Zeilen und Spalten aus den entsprechenden Matrizen rauszulesen und das Ergebnis zu berechnen. Für eine gegebene Matrix

$$\boldsymbol{A} = \begin{bmatrix} A_{11} & A_{12} & A_{11} \\ A_{21} & A_{22} & A_{22} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \tag{2.23}$$

ergibt sich im ersten Schritt:

$$\boldsymbol{A'} = \boldsymbol{P}^{-1} \cdot \boldsymbol{A} \cdot \boldsymbol{P} \tag{2.24}$$

$$\Rightarrow \boldsymbol{A'} = \begin{bmatrix} 1/A_{11} & -1/A_{21} & 0 \\ 0 & 1/A_{21} & 0 \\ 0 & -1/A_{21} & 1/A_{31} \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{12} & A_{11} \\ A_{21} & A_{22} & A_{22} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{11} & 0 \\ 0 & A_{21} & 0 \\ 0 & A_{31} & A_{31} \end{bmatrix} \tag{2.25}$$

$$\Leftrightarrow \boldsymbol{A'} = \begin{bmatrix} 0 & A_{21}(\frac{A_{12}}{A_{11}} - \frac{A_{22}}{A_{21}}) + A_{31}(\frac{A_{13}}{A_{11}} - \frac{A_{23}}{A_{21}}) & A_{31}(\frac{A_{13}}{A_{11}} - \frac{A_{23}}{A_{21}}) \\ A_{11} & A_{11} + A_{22} + \frac{A_{23}A_{31}}{A_{21}} & \frac{A_{23}A_{31}}{A_{21}} \\ 0 & -A_{21}(\frac{A_{22}}{A_{21}} - \frac{A_{32}}{A_{31}}) - A_{31}(\frac{A_{23}}{A_{21}} - \frac{A_{33}}{A_{31}}) & A_{33} - \frac{A_{23}A_{31}}{A_{21}} \end{bmatrix} \tag{2.26}$$

Danach erfolgt der zweite Schritt

$$\boldsymbol{A''} = \boldsymbol{P'}^{-1} \cdot \boldsymbol{A'} \cdot \boldsymbol{P'} \tag{2.27}$$

und der zusätzliche Schlussschritt:

$$\boldsymbol{A}''' = \boldsymbol{P}''^{-1} \cdot \boldsymbol{A}'' \cdot \boldsymbol{P}'' \tag{2.28}$$

Das Ergebnis kann dann aus der letzten Spalte abgelesen werden, wobei in der Subdiagonalen nur Einsen übrig bleiben müssen.

$$\Rightarrow \boldsymbol{A}''' = \begin{bmatrix} 0 & 0 & A_{11}A_{22}A_{33} - A_{11}A_{23}A_{32} - A_{12}A_{21}A_{33} + A_{12}A_{23}A_{31} + A_{13}A_{21}A_{32} - A_{13}A_{22}A_{31} \\ 1 & 0 & A_{12}A_{21} - A_{11}A_{22} - A_{11}A_{33} + A_{13}A_{31} - A_{22}A_{33} + A_{23}A_{32} \\ 0 & 1 & A_{11} + A_{22} + A_{33} \end{bmatrix} \tag{2.29}$$

Wie man sieht, wurde die Spur von $\boldsymbol{A}$ im untersten Eintrag der letzten Spalte bzw. im letzten Hauptdiagonalelement erhalten, während die Determinante (mit entsprechendem Vorzeichen) im obersten Eintrag der letzten Spalte ermittelt wurde.

3 Fazit

Bei der Ermittlung der zusätzlichen Transformationsmatrix in Gleichung 2.20 für den abschließenden Schritt wird klar, dass bei entsprechend großen Matrizen und Einträgen ein numerisches Problem entsteht. Deshalb ist diese Berechnungsmethode in der Praxis zwar für symbolische Kalkulationen nützlich, aber für numerische Berechnungen wenig sinnvoll. Das herkömmliche Danilewski-Verfahren funktioniert etwas eleganter (siehe Code-Beispiel im Anhang 4.3), doch die numerische Instabilität ist auch da hinlänglich bekannt, weshalb es Sinn macht, obwohl es kontraproduktiv erscheint, einen Umweg über eine iterative Berechnung der Eigenwerte zu gehen. Ein modernes Verfahren wäre z.B. der QR-Algorithmus, welcher zwar iterativ arbeitet, aber dafür numerisch wesentlich stabiler ist. Anschließend kann aus den Eigenwerten, die ja die Nullstellen des charakteristischen Polynoms darstellen, die Begleitmatrix bzw. das charakteristische Polynom hergeleitet werden, falls man diese benötigt.

4 Anhang

Die folgenden Code-Beispiele sind die Umsetzung der bilateralen Elimination für die Ermittlung der Begleitmatrix. Sie sollen der Verifizierung und zu Vergleichszwecken dienen.

4.1 Matlab-Code: Begleitmatrix (original)

```
function [B] = mcomp(A)
% Achtung: Für symbolische Berechnung geeignet,
% aber Division durch Null möglich!

% Eingabe:
    % A   % Matrix A

% Ausgabe:
    % B   % Begleitmatrix B

% Algorithmus:
    B = A; n = size(B,1); I = eye(n);
    if isa(B,'sym')
            I = sym(I);
    end
    for j = 1 : n - 1
            x = j + 1;
            P_mult = diag(B(:, j));
            P_add = I; P_add(:, x) = 1;
            P = P_mult * P_add;
            B = P^-1 * B * P;
    end

% Schlussschritt:
    P = I;
    for i = 2 : n
            P(i, i) = P(i - 1, i - 1) * B(i, i - 1);
    end
    B = P^-1 * B * P;
return
```

4.2 Matlab-Code: Begleitmatrix (optimiert)

```
function [B] = mcomp(A)
% Achtung: Für symbolische Berechnung geeignet,
% aber Division durch Null möglich!

% Eingabe:
    % A   % Matrix A

% Ausgabe:
    % B   % Begleitmatrix B

% Algorithmus:
    B = A; n = size(B,1);
    for j = 1 : n - 1
            x = j + 1;
            v = B(:, j);
            h = B(x, :)/v(x);
            s = v * 0;
            for i = 1 : n
                B(:, i) = (B(:, i)./v - h(i)) * v(i);
                B(x, i) = B(x, i) + h(i) * v(i);
                s = s + B(:, i);
            end
            B(:, x) = s;
    end

% Schlussschritt:
    fact = 1;
    for i = 2 : n
            fact = fact * B(i, i - 1);
            B(i, n) = B(i, n)/fact;
            B(i, i - 1) = 1;
    end
    B(:, n) = B(:, n). * fact;
return
```

4.3 Matlab-Code: Begleitmatrix (Danilewski)

```
function [B] = mcomp(A)
% Achtung: Für symbolische Berechnung geeignet,
% aber Division durch Null möglich!

% Eingabe:
    % A    % Matrix A

% Ausgabe:
    % B    % Begleitmatrix B

% Algorithmus:
    B = A; n = size(B,1); I = eye(n);
    if isa(B,'sym')
            I = sym(I);
    end
    for j = 1:n-1
            x = j+1;
            P = I; P(:,x) = B(:,j);
            B = P^-1 * B * P;
    end
return
```

Literatur

[1] Danilewski, A. M.: *On the numerical solution of the secular equation*

[2] Zurmühl, R. / Falk, S.: *Matrizen und ihre Anwendungen*

Anmerkung zur Erstellung

[3] Gemini AI (Google): *Unterstützung zur besseren Strukturierung und Verifizierung*